DESIGN AND DEVELOPMENT OF SOC BASED NETWORK ON CHIP TOPOLOGIES

T. Nagalaxmi[1], E. Sreenivasa Rao[2] and P. Chandrasekhar

[1,3]*Department of Electronics and Communication Engineering, Osmania University, India*

[2]*Department of Electronics and Communication Engineering, Vasavi College of Engineering, India*

Abstract

The submicron technologies are revealing problems in the interconnections. Thus, the performance of the system largely depends on the structure of communication, in particular with regard to the flow, surface area and power consumed. In addition, traditional communications structures, which are generally based on shared buses, are limited in terms of performance. Indeed, they do not support high flow rates and they do not allow many elements to be interconnected, which makes them not very extensible. Based on this observation, several research groups have worked on a new form of interconnection adapted to future even more complex systems on a chip. They proposed the paradigm of networks on chip. In this paper, various SoC based network topologies has been implemented. Among various network topologies, the four well known topologies are selected. These selected four topologies are implemented by using Verilog programming language and corresponding results has been discussed.

Keywords:

Network on Chip, Network Topologies, System on Chip, FPGA

1. INTRODUCTION

The performance of an architecture implemented in a SoC strongly depends on the system of interconnection and communication protocol between computing units. With increasing integration technology, the design of an interconnection system efficient is critical to fully exploit the number and processing power calculation units in the same circuit [1]. In particular, the permanent increase in the densities of logic-based systems reconfigurable with fine grain FPGA type, offers very large possibilities of parallelisation and integration of calculation units, to create interconnection solutions complex between elements. On-chip interconnection systems [2] are generally architectural adaptations. Scaled-down rules of existing larger-scale solutions, such as for example clusters of processors on electronic cards communicating on a shared bus or a network of processors on the same card.

1.1 VARIETY OF INTERCONNECTION SYSTEMS

There are different types of interconnection systems that can be used in a SoC whose point to point, the shared bus or hierarchical, the crossbar and the on-chip network

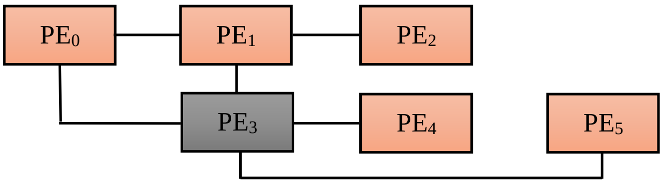
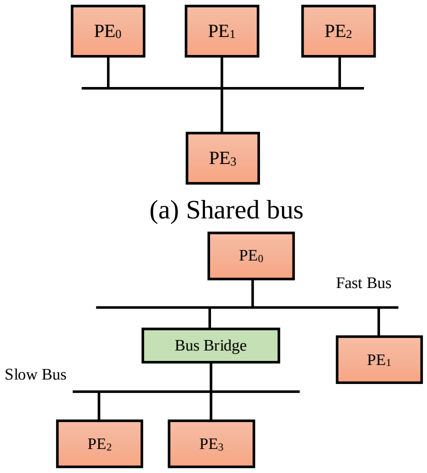


Fig.1. Point-to-point interconnection systems

The point to point approach (Fig.1) is the most direct solution and simpler [3]. It consists in connecting the different computation units of a system with dedicated and exclusive wires for data exchange. Therefore, this approach is efficient for high bandwidth system. It offers a very large possibility of parallelization but implies a weak reuse of calculation units due to the rigidity of the connections. This results in a very low flexibility of communication between computing units. This solution remains however suitable for systems with low number of units. Thus, to evolve a system with this approach requires making connections more complex, by adding increasingly links between units. This method becomes, in a way obvious, difficult to manage with a process of increasing integration of calculation for reasons of physical dependencies between links and synchronization between signals.

The shared bus approach (Fig.2(a)) is a technique widely used for inter-connect computing units [4]. This approach is also suitable for systems with low number of calculation units. Different buses are usually connected. Tees in hierarchical form (Fig.2(b)) by grouping units according to the constraints in bandwidth.



(a) Shared bus

(b) Hierarchical Bus

Fig.2. Bus type interconnection system

A bus is generally made up of data lines, lines control and a referee. This type of approach has the advantage of having a simple work at low cost but is however very limited in terms of performance because this type of communication only allows one module to communicate at a time according to arbitration. This results in very strong data bottleneck formations. Frequent with the increase of connected units.

An expensive approach is to use a crossbar to interconnect the units of calculation. The principle consists in defining a matrix of multiplexers allowing any unit of the system to communicate with another, in the most efficient way possible point-to-point, and thus allowing parallel communications. This approach is very expensive on the surface but suitable for systems with a number reduced calculation units. A compromise also consists in partially achieving this matrix according to communication needs if they are foreseeable. As the architecture of the reconfigurable processor, called Reconfigurable Operators for Multimedia Applications (ROMA), recently proposed by the CEA-LIST [5-6]. This architecture uses a crossbar as an interconnection system between thick-grained computation units and different memory banks.

For a SoC, Usually different types interconnects are combined in a single SoC, using the advantages of each. Point-to-point communications are for example reserved for critical parts in terms of time and shared bus communications are rather dedicated to devices that are slow and require little bandwidth. The use of crossbar in complete partially solves the performance requirements for SoCs [7]. However, these solutions very quickly turn out to be limited in number of calculation units. With increasing integration technologies, it is now possible to implement in a SoC more advanced interconnection systems such as a network on a single chip. The major contributions in this paper as follows: the recent well known network topologies are shortlisted. The four topologies such as mesh, torus, spidergon and binary tree are implemented by using Verilog programming language in Xilinx Vivado tool. The advantages and drawbacks of each topologies are discussed. The corresponding results and comparative analysis of four topologies are reported.

2. NETWORKS ON A CHIP

2.1 INTRODUCTION TO NOC TOPOLOGIES

The first network concepts and prototypes of on-chip communication, called Network-on-Chip (NoC), appeared [8]. According to the definition proposed by Dally, an interconnection network corresponds to a system for transporting data between terminals. The Fig.3 illustrates an example with three terminals T_0, T_1 and T_2.

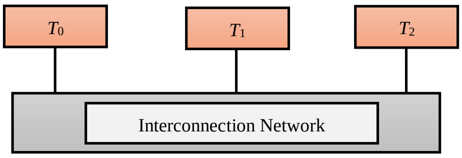


Fig.3. Functional interconnection network with three terminals

The terminals communicate with each other by transmitting messages (data) through network. In our case, a terminal corresponds to a source point or a drop point (sink) data. The network can make several simultaneous connections between terminals thus authorizing several communications in parallel and modifying them at any instant [9]. A network is defined as a system because it is composed of different res-sources: memory, communication channel and a data switching element called router which organize themselves to transmit messages between terminals. We define a channel as a set of wires interconnecting the input and output ports output from routers to transport data.

On the scale of the SoC, let us now consider that each terminal corresponds to a unit of calculation. The interconnection network approach, illustrated in Fig.4, consists of quantize messages to transport them from a source unit to a destination unit, through a network of routers linked by physical communication channels.

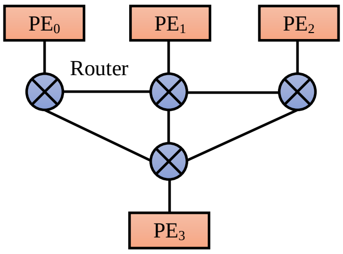


Fig.4. Network type interconnection system

2.2 STRUCTURE OF A MESSAGE IN NOC

Each message, shown in Fig.5, is split into several data packets. The packet is the information unit of the communication network [10]. They are in general composed of a header containing control information and the useful data to transport. In order to be able to be transmitted in the network, these packets are divided into elementary packages called flits (contraction of flow control digits).These flits can be further subdivided into phits (contraction of physical units) corresponding to laying down a unit of data which can be physically transferred through a channel data between two routers in one clock cycle. Performance (i.e. latency, bandwidth, consumption) of a network on a chip depend initially the choice of the granularity of each data unit (message, flits, phits)

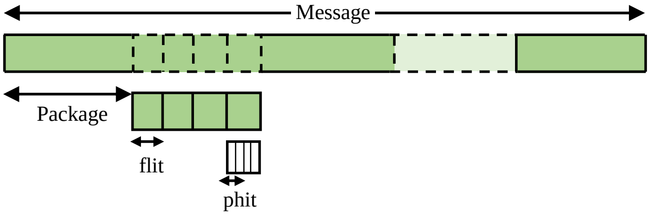


Fig.5. Structure of a message in a NoC network

Numerous studies have been carried out over the past ten years, to carry and evaluate different interconnection solutions of multiprocessor systems towards this new approach to communication network. This is the case, for example, for the architecture Chameleon [11] whose communications have been evaluated in a network context of processors on chip.

The specification of a network is based on three essential points: the topology of the network, which specifies the structure of the network, the switching technique which specifies the mechanisms access to resources and the routing algorithm [12] that determines traffic management (dis-data tributivity) in the network. These points are interdependent. For example, the choice of a topology implies a choice of routing algorithms specific to the topology. Thus, all the difficulty for the designer consists in making compromises in terms of consumption, performance, robustness and complexity.

3. NETWORK TOPOLOGY

The terminology of the communication network uses terms defined in the field of graph theory. The topology of the network [13]-[14] corresponds to the organization of the interconnection of units by communication channels. It is defined as a set N of nodes (vertices) connected by a set C of channels (edges). A channel $c_{x,y}$ such that

$$c_{x,y}=(x,y) \in C|x,y \in N \quad (1)$$

Allows to connect a node x to a node y. The topology of a network is thus represented felt by the graph G.

$$G=(N,C) \quad (2)$$

There is no ideal topology for every system and the designer's difficulty is to achieve a set of trade-offs between connectivity, resource cost and also the choices of the granularity of the messages in the network.

At the architectural level, we call node the computation unit (PE) pair with its associated data router [15], as illustrated in Fig.6.

The degree of freedom of a node corresponds to the number of channels allowing a node to communicate with neighboring nodes. It corresponds to the sum of the number of channels of incoming communication with the number of outgoing communication channels [16]. This parameter significantly affects the complexity of the network and the size of the communication.

The distance between the nodes is generally measured in number of hops. More generally, there can be several possible data paths between two nodes x and y. We call $p_{x,y}$ the corresponding data path between x and y to an ordered set of channels. The number of channels contained in this setthen corresponds to the length of the path.

$$p_{x,y}=\{c_0,c_1,\ldots,c_k\} \,|\forall_I \in \mathbb{N}\,, c_i \in C \quad (3)$$

According to their topology, communication networks are classified into three categories: direct, indirect and irregular networks [17]. In the case of a direct network, each node is connected to neighboring nodes by point-to-point links. The best known direct networks are mesh networks (gridor mesh), torus, folded torus and octagon. The mesh topology is the most common for multi-PE systems for networks on a chip. An example of a 3x3 mesh [18] is shown in Fig.6.

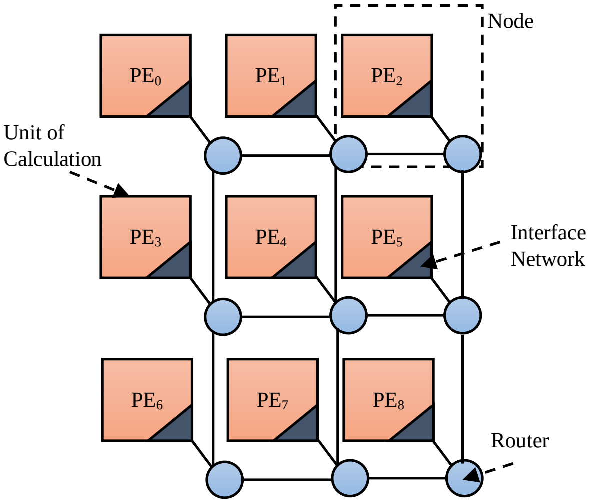


Fig.6. Example of a 3×3 NoC network in grid topology (mesh)

Each node corresponds to a computing unit, a network interface (Network Inter- face (NI)) and a router. The network interface [19] makes it possible to decouple the computing unit from the communication network. It takes care of putting the data issues from the unit in order to transmit it over the network. The router implements the policy routing required to transmit data from one computing unit to another.

Generally, a router is composed of input and output buffers. Output allowing to manage several data packets in parallel, from a crossbar to routing an incoming packet to one of the outputs of the router and a controller allows so much to arbitrate the various referral requests according to the input data and the state of neighboring routers. This global control is governed by a routing algorithm.

In an indirect type network, the routers of each PE are not directly connected but through one or more routers which are connected by point-to-point links [20]. In networks on a chip, tree topologies or butterfly topologies are most used. In a tree-type topology, the compute nodes correspond to the leaf nodes of the tree (ends of the communication network). It is obvious that this topology poses disadvantages in terms of bandwidth and bottleneck of data located mainly at the root node. To overcome this problem, an improvement consists in using a fat tree topology which keeps the same structure that the tree but whose communication channels increase as we get closer to the root node. The objective being to increase the bandwidth progressively to the root node thus reducing data bottlenecks. Finally, irregular networks are generally a mixture of direct and indirect networks. They are used for very specific applications. They are however many more complex in terms of time performance prediction due to the topology irregular. Whatever the chosen topology, a network on a chip is said to be homogeneous or heterogeneous. A network is said to be homogeneous when all the computation units of the network are identical. Otherwise, the network is said to be heterogeneous. The homogeneous approach [21] is the simplest to be implemented. The heterogeneous approach is the most optimized in terms of performance but more complex to implement due to the mixture of unit types. In the embedded field, this second approach is the most relevant. Indeed, the surface being limited, it is often necessary to make heterogeneous computation units coexist in order to ensure complementarity in terms of calculation efficiency.

The implementation of a network on a chip generally involves an obvious additional cost in surface with respect to a point-to-point interconnection. This additional cost is offset by flexibility of parallel communications and better integration of units calculation in the system. A very large number of network-on-chip proposals (NoC) has been studied for several years [22]-[23]. More and more multi-systems processors on a chip then adopted this interconnection solution in areas applications such as telecommunications. Networks on a chip present also a growing interest in the realization of SoC.

4. IMPLEMENTATION OF NETWORK TOPOLOGIES AND RESULTS

In the state of the art there are several network topologies (regular or non-regular), this section describes some topologies of regular type with in detail explanation with support of experimental results:

4.1 MESH NETWORK

This network consists of *m* columns and *n* rows. Routers are located at intersections of two points. The addresses of the routers can be easily defined by XY coordinates in the mesh. The mesh network [24] is the network no longer used thanks to its scalability and flexibility for the application of different routing algorithms. The mesh topology is depicted in Fig.7.

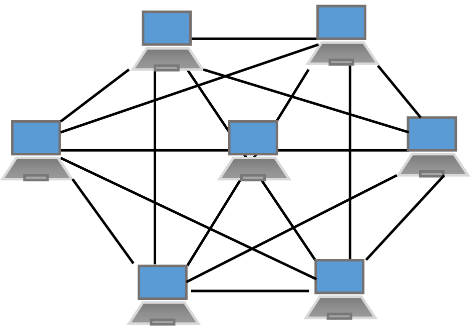
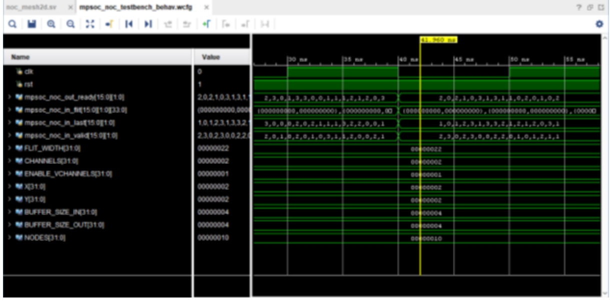


Fig.7. Mesh network topology

The 16×16 mesh network is implemented by using Verilog and data packet transmission on reception is verified. The corresponding results are shown in Fig.8 and device utilization report is shown in Fig.9.



Name	Value
clk	0
rst	1
mpsoc_noc_out_ready[15:0][1:0]	2,0,2,1,0,3,1,3,1,1
mpsoc_noc_in_flit[15:0][1:0][33:0]	(000000000,000
mpsoc_noc_in_last[15:0][1:0]	1,0,1,2,3,1,3,3,2,1
mpsoc_noc_in_valid[15:0][1:0]	2,3,0,2,3,0,0,2,2,0
FLIT_WIDTH[31:0]	00000022
CHANNELS[31:0]	00000002
ENABLE_VCHANNELS[31:0]	00000001
X[31:0]	00000002
Y[31:0]	00000002
BUFFER_SIZE_IN[31:0]	00000004
BUFFER_SIZE_OUT[31:0]	00000004
NODES[31:0]	00000010

Fig.8. Waveforms of Mesh network

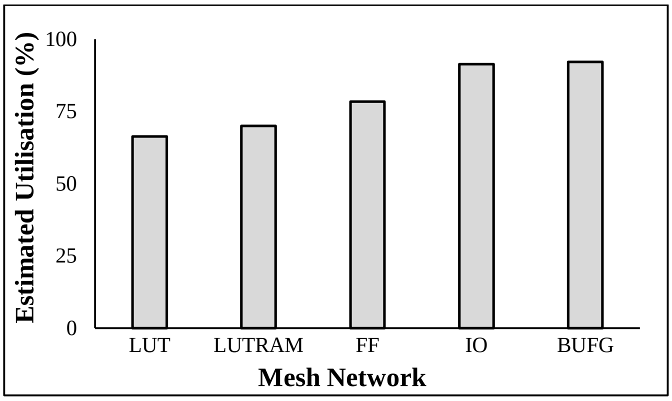


Fig.9. Utilization report of Mesh network

4.2 TORUS NETWORK

A Torus network [25] is an improved version of mesh network. The Torus network structure is simple: it is about a mesh in which the end routers are connected to each other to others in a symmetrical way. The network Torus has better path diversity than that of the mesh network, in return for this network presents a complexity at the level of the implementation of routing algorithms. The pictorial representation of torus network is depicted in Fig.10.

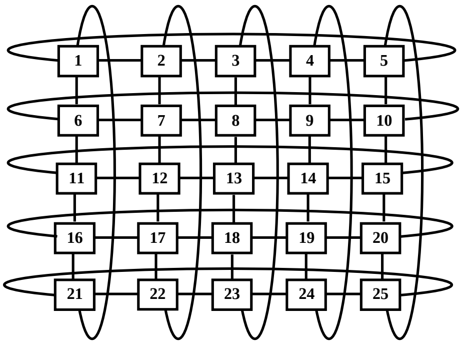
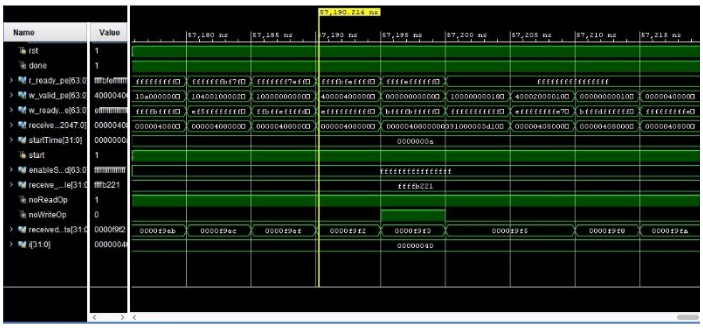


Fig.10. Torus network Topology



Name	Value
rst	1
done	1
r_ready_pe[63:0]	fffbfefffff
w_valid_pe[63:0]	4000040
w_ready...e[63:0]	efffffffff
receive...2047:0]	0000040
startTime[31:0]	0000000
start	1
enableS...d[63:0]	ffffffffff
receive_...le[31:0]	fffb221
noReadOp	1
noWriteOp	0
received...ts[31:0]	0000f9f2
i[31:0]	0000004

Fig.11. Waveforms of Torus network

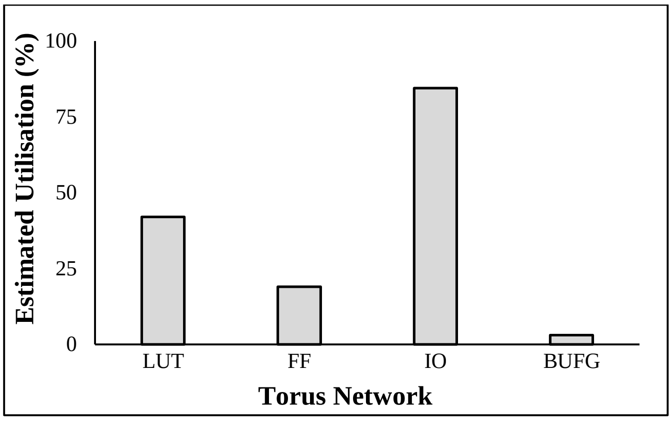


Fig.12. Utilization report of Torus network

In Fig.11, the 16×16 torus network is implemented and tested. These results indicates that, the data packets are effectively transferred and received. The Fig.12 represents utilization report of torus network.

4.3 SPIDERGON

The Spidergon-STNoC [26] is a NoC, which was developed in cooperation with STMicroelectronics and the University of Pisa. It is a NoC for heterogeneous SoCs is developed with many different IPs. The Spidergon-STnoC supports a Variety of different topologies from a ring topology to simple spanning trees up to irregular chordal rings. The NoC also offers special circuits, to enable a mesochronous overall system. As a switching method a wormhole switching is used in the Spidergon-STNoC, which is achieved by two virtual channels is expanded. In addition, router and network interface are supported a packet-based communication. The routing within the NoC is deterministic and encoded in the packet header. This means that packet-based communication has an end-to-end control and thus offers a certain quality of service (GS). However, cycle-specific guarantees about latency and throughput cannot be given. Since delays due to collisions in the routers are not prevented or must be taken into account. The spidergon network representation is reported in Fig.13.

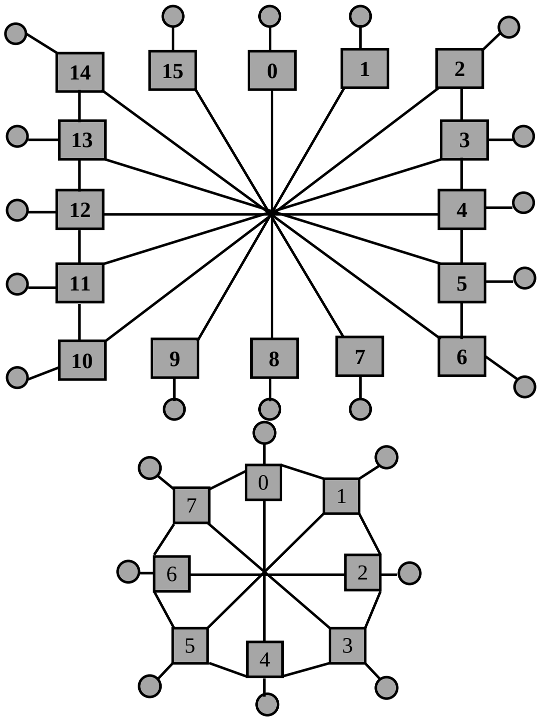
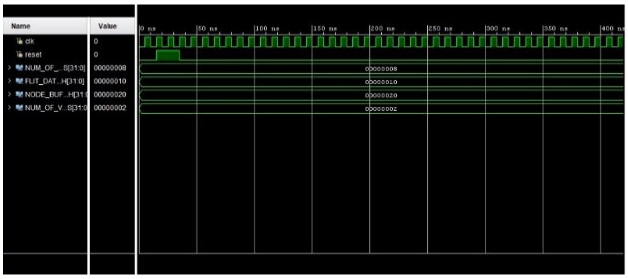


Fig.13. Spidergon network topology



Name	Value
clk	0
reset	0
NUM_OF_...S[31:0]	00000008
FLIT_DAT...H[31:0]	00000010
NODE_BUF...H[31:0	00000020
NUM_OF_V...S[31:0	00000002

Fig.14. Results of Spidergon network

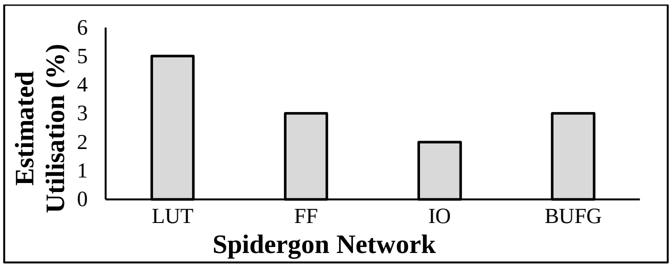


Fig.15. Utilization of Spidergon network

From Fig.14 and Fig.15, the spidergon network requires less number of resources compared with mesh and torus topologies. However the spidergon topology is not suitable for real time implementations.

4.4 TREE NETWORK

The extended tree network is an indirect network (the number of nodes is independent of that of IP cores) where the nodes are routers and the leaves are hearts connected to the network [27]. Routers that are above leaves are called his ancestors and respectively the leaves which are below of their ancestor are called his descendants. In the tree topology each node has multiple ancestors which mean that there are many alternate paths between nodes and that we have a router with a high degree Tree network expanded in butterfly. The tree network is represented in Fig.16.

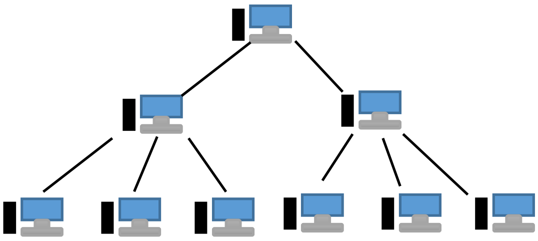
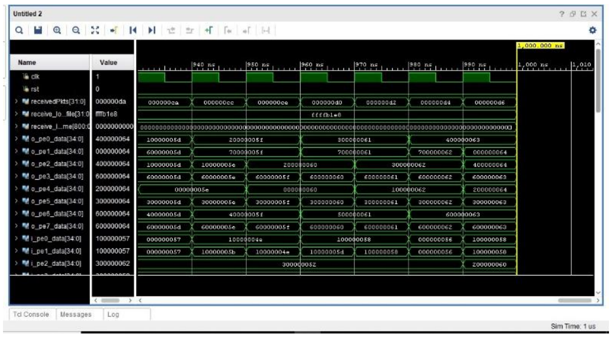


Fig.16. Tree network topology

The butterfly flat-tree topology [28] is used to have a lower degree than topology flat-tree. However, this topology increases the interconnection complexity and the cost surface due to its higher shaft height high. Indeed for this topology each node has two ancestors and four descending nodes. To point out the nodes, coordinates (*L*,*P*) are given to each node where *L* indicates the level of the node, and *P* indicates its position within this level. Address of lowest level is zero and the addresses of nodes are defined from 0 to (*N*-1). There are *N*/4 nodes at the first level and *N*/(2*j*+1) nodes for the j^{th} level. The number of switches at each level is divided by 2.


Untitled 2
940 ns
950 ns
960 ns
970 ns
980 ns
990 ns
1,000 ns
1,010
1,000.000 ns
000000ca 000000cc 000000ce 000000d0 000000d2 000000d4 000000d6
ffffb1e8
0000000000000000000000000000000000000000000000000000000000000000000000000000000
10000005d 20000005f 300000061 400000063
60000005d 70000005f 700000061 700000062 000000064
10000005d 10000005e 200000060 300000062 400000064
60000005d 60000005e 60000005f 600000060 600000061 600000062 600000063
00000005e 000000060 100000062 200000064
30000005d 30000005e 30000005f 300000060 300000061 300000062 300000063
40000005d 40000005f 500000061 600000063
60000005d 60000005e 60000005f 600000060 600000061 600000062 600000063
000000057 10000004e 100000058 000000056 100000058
000000057 10000005b 10000004e 10000005d 100000058 000000056 100000058
300000052 200000060
Tcl Console
Messages
Log
Sim Time: 1 us

Name	Value
clk	1
rst	0
receivedPkts[31:0]	000000da
receive_lo...file[31:0	ffffb1e8
receive_l...me[800:0	0000000000
o_pe0_data[34:0]	400000064
o_pe1_data[34:0]	000000064
o_pe2_data[34:0]	400000064
o_pe3_data[34:0]	600000064
o_pe4_data[34:0]	200000064
o_pe5_data[34:0]	300000064
o_pe6_data[34:0]	600000064
o_pe7_data[34:0]	600000064
i_pe0_data[34:0]	100000057
i_pe1_data[34:0]	100000057
i_pe2_data[34:0]	300000062

Fig.17. Wave forms of synchronous binary tree network

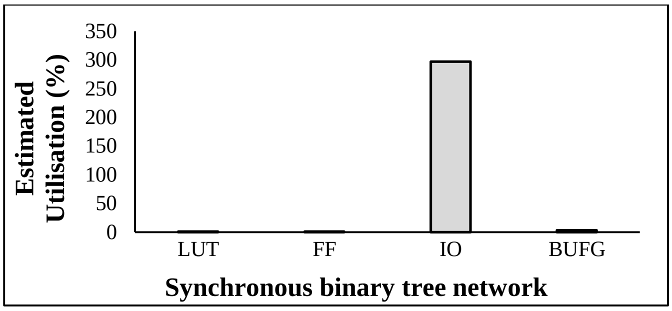


Fig.18. Resources utilization of synchronous binary tree

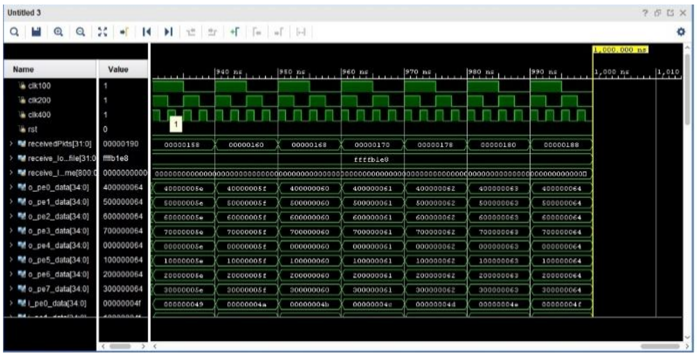

Untitled 3
Name
Value
clk100 1
clk200 1
clk400 1
rst 0
receivedPkts[31:0] 00000190
receive_lo...file[31:0] ffffb1e8
receive_l...me[800:0] 0000000000
o_pe0_data[34:0] 400000064
o_pe1_data[34:0] 500000064
o_pe2_data[34:0] 600000064
o_pe3_data[34:0] 700000064
o_pe4_data[34:0] 000000064
o_pe5_data[34:0] 100000064
o_pe6_data[34:0] 200000064
o_pe7_data[34:0] 300000064
i_pe0_data[34:0] 00000004f
940 ns
950 ns
960 ns
970 ns
980 ns
990 ns
1,000 ns
1,010
1,000.000 ns

Fig.19. Wave forms of Asynchronous binary tree network

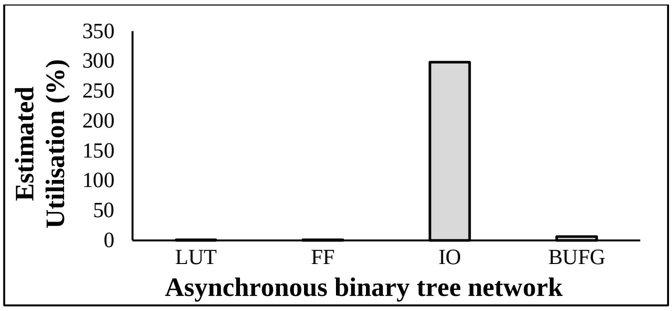


Fig.20. Utilization of Asynchronous binary tree network

The binary tree network implementation [29] is segregated in two modes which are synchronous and asynchronous modes. In synchronous mode the single clock is used for each node. The results and resource utilization of synchronous binary tree network is shown in Fig.17 and Fig.18 respectively. In asynchronous mode, the master clock divided into multiple clocks for each node in the network. The corresponding asynchronous binary tree network results are reported in Fig.19. The resource utilization of asynchronous binary tree network is shown in Fig.20. The comparison results of resource utilization of four topologies reported in Table.1.

Table.1. Comparison result for Estimation utilization

	Mesh (%)	**Torus (%)**	**Spidergon (%)**	**Tree**	
				Sync (%)	**Async (%)**
LUT	313	42	5	1	1
FF	90	19	3	1	1
IO	4481	8449	2	297	298
BUFG	3	3	3	3	6

5. CONCLUSIONS

The design of new Systems on Chip focuses on the structure communication which, like IPs, must be able to be reusable and cope with technological developments. To cope with current developments, new communication structures will have to first, be flexible. In this paper four well known network topologies are implemented which are mesh, torus, spidergon and binary tree. All four topologies are successfully implemented and resource utilizations are reported. Among these four topologies the mesh and torus topologies are suitable for SoC in real time implementation scenario.

REFERENCES

[1] Roberto Interdonato, Martin Atzmueller, Sabrina Gaito, Rushed Kanawati, Christine Largeron and Alessandra Sala, “Feature-Rich Networks: Going Beyond Complex Network Topologies”, *Applied Network Science*, Vol. 4, No. 1, pp. 1-13, 2019.

[2] N. Prasad, Priyajit Mukherjee, SantanuChattopadhyay and Indrajit Chakrabarti, “Design and Evaluation of ZMesh topology for On-Chip Interconnection Networks”, *Journal of Parallel and Distributed Computing*, Vol. 113, pp. 17-36, 2018.

[3] Alak Majumder, Abir J. Mondal and Bidyut K. Bhattacharyya, “Threshold Adjustment of Receiver Chip to Achieve a Data Rate> 66 Gbit/Sec in Point-to-Point Interconnect”, *Integration*, Vol. 58, pp. 348-355, 2017.

[4] Navaridas, Javier, Jose A. Pascual, Alejandro Erickson, Iain A. Stewart and Mikel Lujan. “INRFlow: An Interconnection Networks Research Flow-Level Simulation Framework”, *Journal of Parallel and Distributed Computing*, Vol. 130, pp. 140-152, 2019.

[5] Erwan Raffin, Ch Wolinski, François Charot, Krzysztof Kuchcinski, Stephane Guyetant, Stephane Chevobbe and Emmanuel Casseau, “Scheduling, Binding and Routing System for a Run-Time Reconfigurable Operator based Multimedia Architecture”, *Proceedings of International Conference on Design and Architectures for Signal and Image Processing*, pp. 168-175, 2010.

[6] Elie Duthoo and Olivier Mesnard, “CEA LIST: Processing Low-Resource Languages for CoNLL 2018”, *Proceedings of International Conference on Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pp. 34-44. 2018.

[7] Stratis Ioannidis and Edmund Yeh, “Jointly Optimal Routing and Caching for Arbitrary Network Topologies”, *IEEE Journal on Selected Areas in Communications*, Vol. 36, No. 6, pp. 1258-1275, 2018.

[8] Travis Boraten and Avinash Karanth Kodi, “Runtime Techniques to Mitigate Soft Errors in Network-on-Chip (NoC) Architectures”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 37, No. 3, pp. 682-695, 2017.

[9] Fawaz Alazemi, ArashAzizimazreah, Bella Bose and Lizhong Chen, “Routerless Network-on-Chip”, *Proceedings of International Conference on High Performance Computer Architecture*, pp. 492-503, 2018.

[10] Eduardo Wachter, Luciano L. Caimi, ViniciusFochi, Daniel Munhoz and Fernando G. Moraes, “BrNoC: A Broadcast NoC for Control Messages in Many-Core Systems”, *Microelectronics Journal*, Vol. 68, pp. 69-77, 2017.

[11] Jagadish B. Kotra, Haibo Zhang, Alaa R. Alameldeen, Chris Wilkerson and Mahmut T. Kandemir, “Chameleon: A Dynamically Reconfigurable Heterogeneous Memory System”, *Proceedings of Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 533-545, 2018.

[12] Qixun Zhang, Menglei Jiang, Zhiyong Feng, Wei Li, Wei Zhang and Miao Pan, “IoT Enabled UAV: Network

Architecture and Routing Algorithm", *IEEE Internet of Things*, Vol. 6, No. 2, pp. 3727-3742, 2019.

[13] Santiago Segarra, Antonio G. Marques, Gonzalo Mateos and Alejandro Ribeiro, "Network Topology Inference from Spectral Templates", *IEEE Transactions on Signal and Information Processing over Networks*, Vol. 3, No. 3, pp. 467-483, 2017.

[14] Hui Jiang, Linjuan Sun, Juan Ran, JianxiaBai and Xiaoye Yang, "Community Detection Based on Individual Topics and Network Topology in Social Networks", *IEEE Access*, Vol. 8, pp. 124414-124423, 2020.

[15] Wai-Xi, Jun Cai, Qing Chun Chen and Yu Wang, "DRL-R: Deep Reinforcement Learning Approach for Intelligent Routing in Software-Defined Data-Center Networks", *Journal of Network and Computer Applications*, Vol. 28, pp. 1-20, 2020.

[16] Roberto Sanchez and Jean Pierre David, "Ultra-Low Latency Communication Channels for FPGA-based HPC Cluster", *Integration*, Vol. 63, pp. 41-55, 2018.

[17] Peter R.Monge, and Noshir S. Contractor, "Emergence of Communication Networks", *The New Handbook of Organizational Communication: Advances in Theory, Research, and Methods*, pp. 440-502, 2001.

[18] Wang Zhang, Ligang Hou, Jinhui Wang, ShuqinGeng and Wuchen Wu, "Comparison Research between XY and Odd-Even Routing Algorithm of a 2-Dimension 3x3 Mesh Topology Network-On-Chip", *Proceedings of Global Congress on Intelligent Systems*, Vol. 3, pp. 329-333, 2009.

[19] Sonal Yadav and Hemangee K. Kapoor, "Lightweight Message Encoding of Power-Gating Controller for On-Time Wakeup of Gated Router in Network-on-Chip", *Proceedings of International Symposium on Embedded Computing and System Design*, pp. 1-6, 2019.

[20] Mohsen Jahanshahi, and Fathollah Bistouni, "Crossbar-Based Interconnection Networks", *Series: Computer Communications and Networks*, Vol. 12, pp. 164-173, 2018.

[21] Ali Shokouhi, Marzieh Badkoobe, Farahnaz Mohanna, Ali Asghar Rahmani Hosseinabadi, and Arun Kumar Sangaiah, "Survey on Clustering in Heterogeneous and Homogeneous Wireless Sensor Networks", *The Journal of Supercomputing*, Vol. 74, No. 1, pp. 277-323, 2018.

[22] Fawaz Alazemi, Arash Azizimazreah, Bella Bose and Lizhong Chen, "Routerless Network-on-Chip", *Proceedings of IEEE International Symposium on High Performance Computer Architecture*, pp. 492-503, 2018.

[23] Kun Chih Chen, Masoumeh Ebrahimi, Ting Yi Wang and Yuch Chi Yang, "NoC-based DNN Accelerator: A Future Design Paradigm", *Proceedings of IEEE/ACM International Symposium on Networks-on-Chip*, pp. 1-8. 2019.

[24] Zhanserik Nurlan, Tamara Zhukabayeva and Mohamed Othman, "Mesh Network Dynamic Routing Protocols", *Proceedings of IEEE International Conference on System Engineering and Technology*, pp. 364-369, 2019.

[25] Akash Punhani, NeetuFaujdar and Sunil Kumar, "Design and Evaluation of Cubic Torus Network-on-Chip Architecture", *International Journal of Innovative Technology and Exploring Engineering*, Vol. 8, No. 6, pp. 1672-1676, 2019.

[26] Miltos D. Grammatikakis, Riccardo Locatelli, Giuseppe Maruccia and Lorenzo Pieralisi, "*Design of Cost-Efficient Interconnect Processing Units: Spiderg on STNoC*", CRC Press, 2020.

[27] Xueye Chen, Zhen Zhang, Dengli Yi and Zengliang Hu, "Numerical Studies on Different Two-Dimensional Micromixers Basing on a Fractal-Like Tree Network", *Microsystem Technologies*, Vol. 23, No. 3, pp. 755-763, 2017.

[28] John Kim, William J. Dally and Dennis Abts, "Flattened Butterfly: A Cost-Efficient Topology for High-Radix Networks", *Proceedings of Annual International Symposium on Computer Architecture*, pp. 126-137, 2007.

[29] Daifeng Zhang, and Haibin Duan, "Switching Topology Approach for UAV Formation based on Binary-Tree Network", *Journal of the Franklin Institute*, Vol. 356, No. 2, pp. 835-859, 2019.